051 使用海龟（turtle）模块绘制圣诞树

* 核心技术

**在实现绘制圣诞树时，需要使用python内置的**turtle模块。**turtle模块是python的内置模块，所以在使用前需要使用import导入该模块。代码如下：**

import **turtle**

**创建游戏窗体可以使用turtle模块中的setup()方法来实现。语法格式如下：**

**turtle.setup(width=\_CFG["width"],height=\_CFG["height"],startx=\_CFG["leftright"],starty=\_CFG["topbottom"])**

**参数说明如下：**

* **width**：表示窗体的宽度，如果是整数，大小以像素为单位，如果是浮动，则为屏幕的一小部分；默认值为屏幕的50%。
* **height：表示窗体的高度，**如果是整数，高度以像素为单位，如果是浮动，则为屏幕的一小部分；默认值为屏幕的75%**。**
* **startx：表示窗体显示在屏幕的横向位置，**如果为正，则以像素为单位从屏幕左边缘开始，如果为负，则从右边缘开始，如果没有，则水平居中窗口**。**
* **starty：表示窗体显示在屏幕的纵向位置，**如果为正，则以像素为单位从屏幕上边缘开始，如果为负，则从下边缘开始，如果没有，则垂直居中窗口**。**

**创建游戏窗体的关键代码如下：**

1. **import** turtle # 导入绘图海龟模块
2. turtle.setup(1200,800,0,0) # 创建窗体大小
3. turtle.mainloop() # 不关闭窗体

**turtle对象中的color()方法，用于设置画笔的颜色。其语法格式如下：**

**turtle.color(\*args)**

**参数说明如下：**

* **args：表示背景需要设置的颜色值，例如直接指定颜色值名称**"orange"**或者"**#FFA500"**以及RGB小数值。**

**实现设置画笔颜色的示例代码如下：**

1. **import** turtle # 导入绘图海龟模块
2. t = turtle.Turtle() # 创建画笔
3. t.color('red') # 设置画笔颜色
4. t.forward(50) # 默认方向移动50，并绘制直线
5. turtle.mainloop() # 不关闭窗体

运行结果如图1所示。



**图1 设置画笔颜色**

**turtle对象中的right()方法，用于设置画笔向右旋转的角度。其语法格式如下：**

**turtle.right(angle)**

**参数说明如下：**

* **angle：表示设置画笔向右旋转的角度值。**

**实现设置画笔向右旋转角度的示例代码如下：**

1. **import** turtle # 导入绘图海龟模块
2. t = turtle.Turtle() # 创建画笔
3. t.color('red') # 设置画笔颜色
4. **for** i **in** range(4):
5. t.forward(50) # 默认方向移动50，并绘制直线
6. t.right(90) # 向右旋转90度
7. turtle.mainloop() # 不关闭窗体

运行结果如图2所示。



**图2 设置画笔旋转角度**

**说明：设置画笔的旋转角度除了使用right()方法设置画笔向右旋转以外，还可以使用left()方法设置画笔向左旋转。**

**turtle对象中的goto()方法，用于设置画笔从当前位置移动至指定坐标的位置，并在起始位置与结束位置绘制一条直线。其语法格式如下：**

**turtle.goto(x,y=None)**

**参数说明如下：**

* **x：表示移动目标位置的x坐标。**
* **y：表示移动目标位置的y坐标。**

**实现移动指定坐标位置并绘制直线的示例代码如下：**

1. **import** turtle # 导入绘图海龟模块
2. t = turtle.Turtle() # 创建画笔
3. t.color('red') # 设置画笔颜色
4. t.goto(-230,145) # 画笔移动指定坐标位置，并绘制直线
5. turtle.mainloop() # 不关闭窗体

运行结果如图3所示。



**图3 移动指定坐标位置并绘制直线**

说明:如果通过up()方法将画笔抬起，此时画笔会**移动至指定坐标的位置，但不会绘制直线。**

**turtle对象中的shape()方法，用于将海龟形状设置为具有给定名称的形状。其语法格式如下：**

turtle.shape(name=None)

参数说明如下：

* name：表示给定名称的形状，如“箭头”、“乌龟”、“圆”、“正方形”、“三角形”。

实现移动绘制多个图形的示例代码如下：

1. **import** turtle #导入turtle模块
2. turtle.setup(300,300) # 设置窗体大小
3. t = turtle.Turtle() # 创建画笔
4. t.shape('triangle') # 设置为三角形
5. t.up() # 抬笔
6. **for** i **in** range(1,5): # 循环遍历4行
7. y = 20\*i # 每行纵向的距离
8. **for** j **in** range(3): # 循环遍历三列，左右各三列中间列重叠
9. x = 20\*j # 横向的距离
10. t.goto(x,-y+60) # 移动至右侧需要绘制的位置
11. t.stamp() # 复制图形，实现绘制右侧图形
12. t.goto(-x,-y+60) # 移动至左侧需要绘制的位置
13. t.stamp() # 复制图形，实现绘制左侧图形
14. t.hideturtle() # 隐藏箭头
15. turtle.mainloop() # 开始循环防止窗口自动关闭

运行结果如图4所示。



图4 移动绘制多个图形

**turtle对象中的write()方法，用于实现在窗体中写入指定的文字内容。其语法格式如下：**

turtle.write(arg, move=False, align="left", font=("Arial", 8, "normal"))

参数说明如下：

* arg：表示需要写入窗体中的文字。
* move：表示是否移动，True表示移动，False表示不移动。
* align：表示对齐方式，参数可以设置为字符类型的“left”“right”或“center”。
* font：表示设置字体参数的元组，元组中三个参数分别表示“字体”“大小”“类型”。

实现在窗体中写入文字的示例代码如下：

1. **import** turtle # 导入绘图海龟模块
2. t = turtle.Turtle() # 创建画笔
3. t.write("不要等待机会，而要创造机会。",font=(u"黑体",20,"normal"),align="center")
4. turtle.mainloop() # 不关闭窗体

运行结果如图5所示。



图5 在窗体中写入文字